Vietnam J Comput Sci (2015) 2:143–148
DOI 10.1007/s40595-015-0039-8

REGULAR PAPER

A novel algorithm applied to filter spam e-mails for iPhone

Zne-Jung Lee · Tsung-Hui Lu · Hsiang Huang

Received: 18 August 2014 / Accepted: 25 January 2015 / Published online: 8 February 2015

Abstract An iPhone is like a portable computer in arithmetic logic capability, memory capacity, and multi-media capability. Many malware targeting iPhone has already emerged and then the situation is getting worse. Accessing e-mails on the internet is not a new capability for iPhone. Other smart phones on the market also provide this capability. The spam e-mails have become a serious problem for iPhone. A novel algorithm, artificial bee-based decision tree (ABBDT) approach, is applied to filter spam e-mails for iPhone in this paper. In the proposed ABBDT approach, decision tree is used to filter spam e-mails for iPhone. In addition, artificial bee algorithm is used to increase the testing accuracy of decision tree. There are total 504 collected e-mails in iPhone dataset and they are categorized into 12 attributes. Another spambase dataset, obtained from UCI repository of machine learning databases, is also used to evaluate the performance of the proposed ABBDT approach. From simulation results, the proposed ABBDT approach outperforms other existing algorithms.

Keywords Spam e-mails · iPhone · Decision tree · Artificial bee · Artificial bee-based decision tree (ABBDT)

Z.-J. Lee (✉)
Department of Information Management, Huafan University, Taipei, Taiwan
e-mail: johnlee@cc.hfu.edu.tw

T.-H. Lu · H. Huang
Department of Mechatronic Engineering, Huafan University, Taipei, Taiwan

1 Introduction

Apple's iPhone was released on June 29, 2007. Today iPhone has evolved and experienced an immense popularity due to its ability to provide a wide variety of services to users. Thereafter, iPhone is inevitably becoming the hot targets of hackers and there are many malicious programs targeting iPhone [1]. Two known root exploits on iPhone are: Libtiff and SMS fuzzing [2]. The attackers could use these exploits to steal personal data from iPhone. For Libtiff, discovered by Ormandy, it opens a potential vulnerability that could be exploited when SSH is actively running [3–6]. Rick Farrow demonstrated how a maliciously crafted TIFF can be opened and lead to arbitrary code execution [7]. The SMS fuzzing is another iPhone exploit that can allow a hacker to control the iPhone through SMS messages [5,8]. The first worm, known as iKee, was developed by a 21-year-old Australian hacker named Ashley Towns [9]. This worm could change iPhone's wallpaper to a photograph of the British 1980s pop singer named Rick Astley. After two weeks, a new malware named iKee.B was spotted by XS4ALL across almost Europe [9]. The iSAM is an iPhone stealth airborne malware incorporated six different malware mechanisms [10]. It could connect back to the bot server to update its programming logic or to obey commands and unleash a synchronized attack. The iPhone has the ability to zoom in/out and view maps. A lot of widgets with finger touches to the screen are also available for iPhone. Thereafter, the iPhone can easily access e-mails on the internet and store personal data. The spam e-mails are sent to users' mailbox without their permission. The over-abundant of spam e-mails not only affects the network bandwidth, but also becomes the hotbeds of malicious programs in information security [11]. It is an important issue for users of iPhone to filter spam e-mails and then prevent the leakage of personal data.

Traditionally, machine learning techniques formalize a problem of clustering of spam message collection through the objective function. The objective function is a maximization of similarity between messages in clusters, which is defined by k-nearest neighbor (kNN) algorithm. Genetic algorithm including penalty function for solving clustering problem is also proposed [12]. Unfortunately, above approaches do not provide good enough performance to filter spam e-mails for iPhone. In this paper, an artificial bee-based decision tree (ABBDT) is applied to filter spam e-mails for iPhone. In the proposed approach, decision tree is used to filter spam e-mails. In addition, artificial bee algorithm is used to ameliorate the testing accuracy of decision tree.

The remainder of this paper is organized as follows. The proposed approach is based on decision tree and artificial bee colony. Section 2 first introduces decision tree and artificial bee colony. Then, Sect. 3 introduces the proposed ABBDT approach to filter spam e-mails. Experimental results are compared with those of existing algorithms in Sect. 4. Conclusions and future work are finally drawn in Sect. 5.

2 The introduction of decision tree and artificial bee colony

The proposed ABBDT approach is based on decision tree and artificial bee colony (ABC). In this section, the brief descriptions of decision tree and artificial bee colony are introduced.

For artificial bee colony algorithm, proposed by Karaboga in 2005, simulates the foraging behavior of a bee colony into three groups of bees: employed bees (forager bees), onlooker bees (observer bees) and scouts [13]. ABC algorithms have been applied in many applications [14–21]. The ABC algorithm starts with randomly produced initial food sources that correspond to the solutions for employed bees. In the ABC algorithm, each food source has only one employed bee. Employed bees investigate the food source and share their food information with onlooker bees in a hive. The higher quality of food source, the larger probability will be selected by onlooker bees. The employed bee of a discarded food source becomes a scout bee for searching for new food source. For decision tree (DT) learning algorithm, proposed by Quinlan, is a tree-like rule induction approach that the representing rules could be easily understood [22]. DT uses the partition information entropy minimization to recursively partition the data set into smaller subdivisions, and then generates a tree structure. This tree-like structure is composed of a root node (formed from all of the data), a set of internal nodes (splits), and a set of leaf nodes. A decision tree can be used to classify patterns by starting at the root node of the tree and moving through it until a leaf node has met [23–26].

3 The proposed ABBDT approach

The operating system of iPhone, named as iOS, is defined at the WWDC conference in 2010 [27]. The iOS architecture is divided into core operating system layer, core service layer, media layer, and cocoa touch layer. Each layer provides programming frameworks for the development of applications that run on top of the underlying hardware. The iOS architecture is shown in Fig. 1. Using tools, it is easy to collect e-mails stored at the path of "/var/mobile/library/mail" [28,29].

The flow chart of the proposed ABBDT approach is shown in Fig. 2.

In Fig. 2, the dataset is first pre-processed as training and testing data and then the initial solutions are randomly generated. There are 12 attributes in the iPhone e-mails dataset as shown in Table 1. Some of these attributes are also important for spam e-mails in computers [30].

The solution is represented as 12 attributes followed with 2 variables, *MCs* and *CF* as shown in Fig. 3. The initial

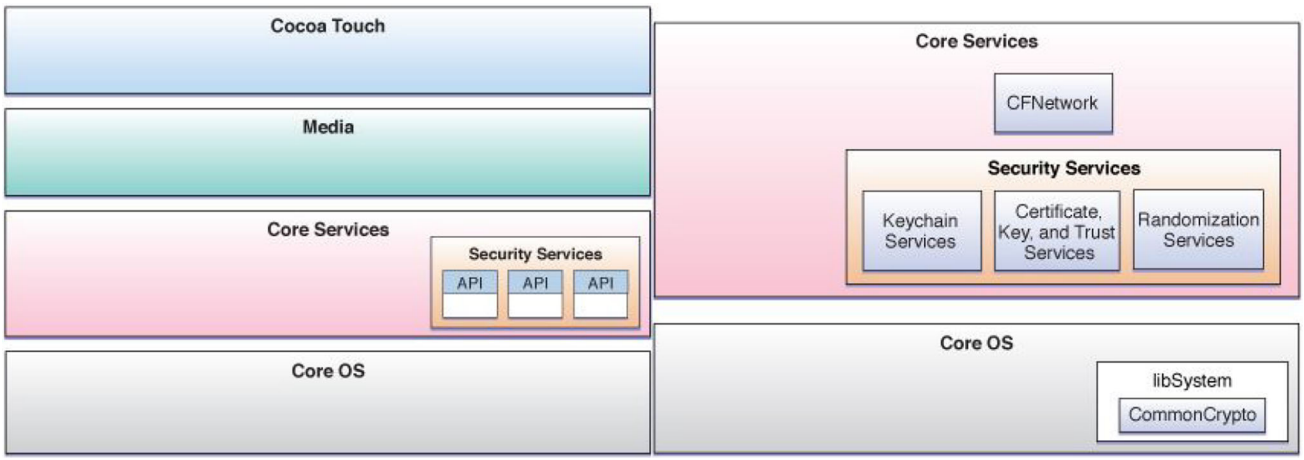

Fig. 1 iOS system architecture

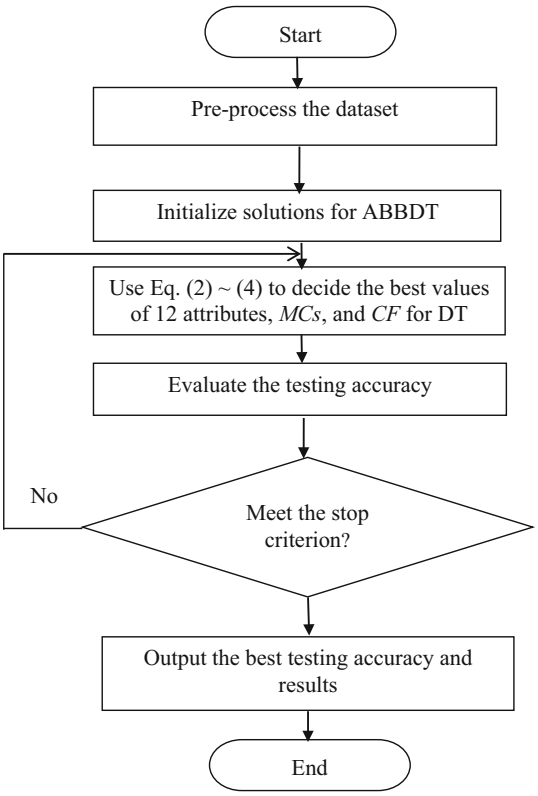

Fig. 2 The flow chart of the proposed algorithm

population of solutions is defined as the number of β in the D-dimensional food sources.

$$F(\boldsymbol{X}_i), \quad X_i \in R^D, \ i \in \{1, 2, 3, \ldots, \beta\} \tag{1}$$

where $\boldsymbol{X}_i = [x_{i1}, x_{i2}, \ldots, x_{iD}]$ is the position of the ith food source and $F(\boldsymbol{X}_i)$ is the object function which represents the quality of the ith food source. To update a feasible food source (solution) position $\boldsymbol{V}_i = [v_{i1}, v_{i2}, \ldots, v_{iD}]$ from the old one $\boldsymbol{X}_i$, the ABC algorithm uses Eq. (2) as follow.

$$v_{ij} = x_{ij} + \varphi_{ij}(x_{ij} - x_{kj}) \tag{2}$$

In Eq. (2), v_{ij} is a new feasible solution, $k \in \{1, 2, 3, \ldots, \beta\}$ and $j \in \{1, 2, 3, \ldots, D\}$ are randomly chosen indexes, k has to be different from j, and φ_{ij} is a random number in the range $[-1, 1]$. After all employed bees complete their searches, they share their information related to the nectar amounts and food sources positions with the onlooker bees on the dance area. An onlooker bee evaluates the nectar information taken from all employed bees. Additionally, the probability for an onlooker bee chooses a food source is defined as Eq. (3).

$$P_i = F(X_i) / \sum_{k=1}^{S} F(X_k) \tag{3}$$

For the food source, its intake performance is defined as F/T where F is the amount of nectar and T is the time spent at the food source [20,31]. If a food source cannot be further improved through a predetermined number of iterations, the food source is assumed to be abandoned, and the corresponding employed bee becomes a scout bee. The new random position chosen by the scout bee is described as follows.

$$x_{ij} = x_j^{\min} + \emptyset_{ij} * (x_j^{\max} - x_j^{\min}) \tag{4}$$

where $x_j^{\min}$ is the lower bound, $x_j^{\max}$ is the upper bound of the food source position in dimension j, and $\emptyset_{ij}$ is a random number in the range [0, 1]. Thereafter, Eqs. (2)–(4) are used to decide the best values of 12 attributes, *MCs*, and *CF* for DT. The values of 12 attributes range from 0 to 1. The corresponding attribute is selected if its value is less than or equal to 0.5. On the other hand, the corresponding attribute is not selected if its value is greater than 0.5. The values of

Table 1 The 12 attributes in the dataset

Number	Attribute	Data type	Example
1	E-mail address is empty	Empty(), Non-empty()	E-mail address is empty or not
2	E-mail domain is empty	Empty(), Non-empty()	E-mail domain is empty or not
3	Account is empty	Empty(), Non-empty()	Account is empty or not
4	Header field is empty	Empty(), Non-empty()	The e-mail should have information in the header filed
5	Special symbol	Appear(), Non-appear()	In the account only appears two kind of marks "." and "_", but there are other special symbols
6	Without @	Yes(), No()	There is no @ in the e-mail address
7	Domain format is error	Yes(), No()	E-mail domain format is error
8	Strange format	Yes(), No()	Time and date have strange values
9	Multi-forward	Continuous	E-mail is forwarded too many times
10	Meaningless subject	Yes(), No()	The subject is meaningless or has error code
11	Duplicate header	Yes(), No()	Duplicate names of header appear in the e-mail
12	Spam e-mail	Yes(), No()	Whether the e-mail is spam or not

Fig. 3 The representation of solution for ABBDT

Attribute #1	Attribute#2	…	Attribute#11	Attribute#12	*M*	*CF*

MCs and *CF* are varied between 1 and 100. In the proposed ABBDT approach, it could select the best subset of attributes to maximize the testing accuracy. When applied to the set of train patterns, Info(S) measures the average amount of information needed to identify the class of the pattern S.

$$\text{Info}(S) = -\sum_{j=1}^{k} \left\{ \left[\text{freq}\left(C_j, \frac{S}{|S|}\right) \right] \log_2 \left[\text{freq}\left(C_j, \frac{S}{|S|}\right) \right] \right\} \quad (5)$$

where $|S|$ is the number of cases in the training set. C_j is a class for $j = 1, 2, \ldots, k$ where k is the number of classes and $\text{freq}(C_j, \frac{S}{|S|})$ is the number of cases included in C_j. To consider the expected information value $\text{Info}_x(S)$ for attribute X to the partition S, it can be stated as:

$$\text{Info}_x(S) = -\sum_{j=1}^{n} \left\{ \left[\left(\frac{|S_j|}{|S|} \right) \text{Info}(S_j) \right] \right\} \quad (6)$$

where n is the number of output for the attribute X, S_j is a subset of S corresponding to the jth output and $|S_j|$ is the number of cases of the subset S_j. The information gain according to attribute X is shown as

$$\text{Gain}(X) = \text{Info}(S) - \text{Info}_x(S) \quad (7)$$

Then, the potential information SplitInfo(X) generated by dividing S into n subsets is defined as.

$$\text{SplitInfo}(X) = -\sum_{j=1}^{n} \left\{ \left[\left(\frac{|S_j|}{|S|} \right) \log_2 \left(\frac{|S_j|}{|S|} \right) \right] \right\} \quad (8)$$

Finally, the gain ratio GainRatio(X) is calculated as

$$\text{GainRatio}(X) = \text{Gain}(X)/\text{SplitInfo}(X) \quad (9)$$

where the GainRatio(X) represents the quantity of information provided by X in the training set, and the attribute with the highest GainRatio(X) is taken as the root of the decision tree. The proposed ABBDT approach is repeated until the stop criterion has met. Finally, the best testing accuracy and filtered e-mails are reported. The pseudocode of the proposed approach is listed as follow.

```
Procedure: ABBDT
  Begin
    t ← 0;
    Initialize solutions;
    While (not match the termination conditions) do
        Use Eq. (2) ~ (4) to decide the best values of attributes, MCs, and CF;
        Apply Eq. (5)~(9) to obtain the testing accuracy;
        Evaluate the testing accuracy;
        t ← t+1;
    End
    Output the best testing accuracy and filtered e-mails;
  End
```

4 Experimental results

In the proposed ABBDT approach, maximum number of cycles was taken as 1,000. The percentage of onlooker bees was 50 % of the colony, the employed bees were 50 % of the colony and the number of scout bees was selected to be at most one for each cycle. In ABBDT, the number of onlooker bees is taken equal to the number of employed bees [31]. In this paper, the simulation results are compared with DT, back-propagation network (BPN), and support vector machine (SVM). BPN is the most widely used neural network model, and its network behavior is determined on the basis of input–output learning pairs [32,33]. SVM is a learning system proposed by Vapnik that uses a hypothesis space of linear function in a high-dimensional feature space [34]. The k-nearest neighbor algorithm (kNN) is a method for classifying objects based on closest training examples in an n-dimensional pattern space. When given an unknown tuple the classifier searches the pattern space for the k training tuples that are closest to the unknown tuple. These k training tuple are the k nearest neighbor of the unknown tuple [35]. It is noted that the values of parameters for compared approaches are set as the same values as the proposed ABBDT approach. To filter e-mails for iPhone, there are total 504 e-mails in the dataset. In this dataset, 304 e-mails are normal and others are spam e-mails. In this paper, 173 e-mails (normal and spam e-mails) are randomly selected as testing data and others are training data. For the dataset, there are 95 spam e-mails which are correctly filtered as spam e-mails (true negative) and 69 normal e-mails which are also correctly filtered as normal e-mails (true positive). The testing accuracy of the proposed ABBDT approach for this dataset is 94.8 %. The result is shown in Table 2. From Table 2, the proposed ABBDT approach has the best performance among these compared algorithms.

Furthermore, another spambase dataset obtained from UCI repository is used to evaluate the performance for

Table 2 The testing accuracy for iPhone e-mails dataset

Approach	Accuracy (%)
DT	91.8
SVM	90.6
BPN	88.1
*k*NN [36]	89.2
The propose ABBDT approach	94.8

Table 3 The testing accuracy for spambase dataset

Approach	Accuracy (%)
DT	92.6
SVM	91.5
BPN	90.4
*k*NN [36]	90.8
The propose ABBDT approach	93.7

the proposed ABBDT approach [36]. There are 4,601 instances with 57 attributes for spambase dataset. The definitions of the attributes are: (1) 48 continuous real [0, 100] attributes of type word_freq_word. A "word" means any string of alphanumeric characters bounded by non-alphanumeric characters or end-of-string. (2) 6 continuous real [0, 100] attributes of type char_freq_char (3) 1 continuous integer [1, . . .] attribute of type capital_run_length_longest. (4) 1 continuous integer [1, . . .] attribute of type capital_run_length_total. (5) 1 nominal {0, 1} class attribute of type spam [24]. In spambase dataset, 1,000 e-mails are randomly selected as testing data and others are training data. The testing accuracy of spambase dataset for the proposed ABBDT approach is 93.7 % as shown in Table 3.

Clearly, the proposed approach outperforms other existing algorithms. It is easy to see that the proposed ABBDT approach outperforms DT, SVM, *k*NN and BPN, individually.

5 Conclusions and future work

In this paper, artificial bee-based decision tree (ABBDT) approach is applied to filter spam e-mails for iPhone. The dataset of iPhone is divided into 12 attributes and there are total 504 e-mails in this dataset. For spambase dataset, there are 4,601 instances with 57 attributes. A comparison of the obtained results with those of other approaches demonstrates that the proposed ABBDT approach improves the testing accuracy for both datasets. The proposed ABBDT approach was applied to effectively find better values of parameters. Thereafter, it ameliorates the overall outcomes of testing accuracy. From simulation results, the testing accuracy is 94.8 % for iPhone dataset as shown in Table 2. In Table 3, the testing accuracy is 93.7 % for spambase dataset. It indeed shows that the proposed ABBDT approach outperforms other approaches. In the future work, it could add more attributes and apply the proposed approach to build an APP for iPhone.

Acknowledgments The authors would like to thank the National Science Council of the Republic of China, Taiwan for financially supporting this research under Contract No. NSC 101-2221-E-211-010, NSC 102-2632-E-211-001-MY3, and MOST 103-2221-E-211-009.

References

1. Gilbert, P., Chun, B.G., Cox, L., Jung, J.: Automating privacy testing of smartphone applications. Technical Report CS-2011-02, Duke University (2011)
2. Cedric, H., Sigwald, J.: iPhone security model and vulnerabilities. France Sogeti, ESEC (2010)
3. Salerno, S., Ameya, S., Shambhu, U.: Exploration of attacks on current generation smartphones. Procedia Comput. Sci. **5**, 546–553 (2011)
4. Apple iPod touch/iPhone TIFF Image Processing Vulnerability. http://secunia.com/advisories/27213/
5. Rouf, I., Miller, R., Mustafa, H., Taylor, T., Oh, S., Xu, W., Gruteser, M., Trappe, W., Seskar, I.: Security and privacy vulnerabilities of in-car wireless networks: a tire pressure monitoring system case study. In: Goldberg, I. (ed.) USENIX Security 2010, pp. 323–338 (2010)
6. Salerno, S., Ameya, S., Shambhu, U.: Exploration of attacks on current generation smartphones. Procedia Comput. Sci. **5**, 546–553 (2011)
7. Rick, F.: Metasploit and my iphone security interview. http://www.roughlydrafted.com/2007/11/20/unwiredrickfarrowmetasploitandmyiphonesecurityinterview/
8. Hua, J., Kouichi, S.: A sms-based mobile botnet using flooding algorithm. In: Information Security Theory and Practice. Security and Privacy of Mobile Devices in Wireless Communication. Springer, Berlin, Heidelberg, pp. 264–279 (2011)
9. Porras, P., Hassen, S., Vinod, Y.: An analysis of the iKee.B iPhone botnet. In: Security and Privacy in Mobile Information and Communication Systems. Springer, Berlin, Heidelberg, pp. 141–152 (2010)
10. Damopoulos, D., Georgios, K., Stefanos, G.: iSAM: an iPhone stealth airborne malware. In: Future Challenges in Security and Privacy for Academia and Industry. Springer, Berlin, Heidelberg, pp. 17–28 (2011)
11. Brad, M.: Spam filtering using neural networks (2002). http://www.umr.edu/~bmartin/378Project
12. Youn, S., McLeod, D.: A comparative study for email classification. In: Advances and Innovations in Systems, Computing Sciences and Software Engineering, pp. 387–391 (2007)
13. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical Report-TR06. Erciyes University, Engineering Faculty, Computer Engineering Department, Turkey (2005)
14. Dervis, K., Bahriye, A.: A modified artificial bee colony (ABC) algorithm for constrained optimization problems. Appl. Soft Comput. **11**, 3021–3031 (2011)

15. Gao, W.-F., Liu, S.-Y.: A modified artificial bee colony algorithm. Comput. Oper. Res. **39**(3), 687–697 (2012)
16. Karaboga, D., Ozturk, C.: A novel clustering approach: artificial bee colony (ABC) algorithm. Appl. Soft Comput. **11**(1), 652–657 (2011)
17. Karaboga, D., Akay, B.: Artificial bee colony (ABC), harmony search and bees algorithms on numerical optimization. In: Innovative Production Machines and Systems Virtual Conference (2009)
18. Baykasoglu, A., Ozbakir, L., Tapkan, P.: Artificial bee colony algorithm and its application to generalized assignment problem. In: Swarm Intelligence: Focus on Ant and Particle Swarm Optimization, pp. 113–144 (2007)
19. Gao, W., Liu, S.: Improved artificial bee colony algorithm for global optimization. Inf. Process. Lett. **111**(17), 871–882 (2011)
20. Banharnsakun, A., Achalakul, T., Sirinaovakul, B.: The best-so-far selection in artificial bee colony algorithm. Appl. Soft Comput. **11**(1), 2888–2901 (2011)
21. Yang, X.-S.: Nature-Inspired Metaheuristic Algorithms, 2nd edn. Luniver Press, Bristol (2010)
22. Kim, H., Koehler, G.J.: Theory and practice of decision tree induction. Omega **23**, 637–652 (1995)
23. Quinlan, J.R.: Induction of decision trees. Mach. Learn., **1**, 81–106 (1986)
24. Quinlan, J.R.: Decision trees as probabilistic classifiers. In: Proceedings of 4th International Workshop Machine Learning, pp. 31–37. Irvine, California (1987)
25. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann, San Francisco (1993)
26. Quinlan, J.R.: Improved use of continuous attributes in C4.5. J. Artif. Intell. Res. **4**, 77–90 (1996)
27. Whalen, S.: Forensic Solutions for the Digital World Forward Discovery, USA (2008)
28. Bader, M., Baggili, I.: iPhone 3GS Forensics: Logical Analysis using Apple iTunes Backup Utility. Zayed University, Abu Dhabi (2010)
29. Chiou, C.-T.: Design and implementation of digital forensic system—a case study of iPhone smart phone. Master Thesis, Huafan University Taiwan (2011)
30. Ying, K.-C., Lin, S.-W., Lee, Z.-J., Lin, Y.-T.: An ensemble approach applied to classify spam e-mails. Exp. Syst. Appl. **37**(3), 2197–2201 (2010)
31. Karaboga, D., Basturk, B.: On the performance of artificial bee colony (ABC) algorithm. Appl. Soft Comput. **8**, 687–697 (2008)
32. Hong, S.J., May, G.S., Park, D.C.: Neural network modeling of reactive ion etching using optical emission spectroscopy data. IEEE Trans. Semicond. Manuf. **16**, 598–608 (2003)
33. Khaw, J.F.C., Lim, B.S., Lim, L.E.N.: Optimal design of neural networks using the Taguchi method. Neurocomputing **7**, 225–245 (1995)
34. Vapnik, V.N.: Statistical Learning Theory. Wiley, New York (1998)
35. Geetha Ramani, R., Sivagami, G.: Parkinson disease classification using data mining algorithms. Int. J. Comput. Appl. **32**(8), 17–22 (2011)
36. Blake, C., Keogh, E, Merz, C.J.: UCI repository of machine learning databases. Department of Information and Computer Science, University of California, Irvine, CA (1998). http://www.ics.uci.edu/~mlearn/MLRepository.html